COMMUNICATION

6(17), October 1, 2013

ISSN 2278–5469 EISSN 2278–5450

Discovery

Distributed shared memory - an overview

Shahbaz Ali Khan☼**, Harish R, Shokat Ali, Rajat, Vaibhav Jain, Nitish Raj**

CSE department, Dronacharya College of engineering, Gurgaon, Haryana-06, India

☼**Corresponding Author:** CSE department, Dronacharya College of engineering, Gurgaon, Haryana-06, India, E-Mail: shahbazalikhan@outlook.com

Publication History
Received: 17 August 2013
Accepted: 26 September 2013
Published: 1 October 2013

Citation
Shahbaz Ali Khan, Harish R, Shokat Ali, Rajat, Vaibhav Jain, Nitish Raj. Distributed shared memory - an overview. *Discovery*, 2013, 6(17), 16-20

ABSTRACT

The intention of this paper is to provide an overview on the subject of Distributed shared memory. Distributed shared memory (DSM) systems represent a successful hybrid of two parallel computer classes: shared memory multiprocessors and distributed computer systems. They provide the shared memory abstraction in systems with physically distributed memories, and consequently combine the advantages of both approaches. Because of that, the concept of distributed shared memory is recognized as one of the most attractive approaches for building large-scale, high-performance multiprocessor systems. The increasing importance of this subject imposes the need for its thorough understanding. Problems in this area logically resemble the cache coherence-related problems in shared memory multiprocessors.

Keywords: shared memory, consistency, coherence

Abbreviations: DSM- Distributed shared memory, VM- Virtual Memory

1. INTRODUCTION

Traditionally, shared memory and message passing have been the two programming models for interprocess communication and synchronization in computations performed on a distributed system. Message passing has been the preferred way of handling interprocess communication in loosely-coupled systems, because the computers forming a distributed system donot share physical memory. The message passing model is characterized by data movement among cooperating processes as they communicate and synchronize by sending and receiving messages. In contrast, tightly-coupled processors primarily use shared memory model since it provides direct support for data sharing.

What

- The distributed shared memory (DSM) implements the shared memory model in distributed systems, which have no physical shared memory
- The shared memory model provides a virtual address space shared between all nodes
- The overcome the high cost of communication in distributed systems, DSM systems move data to the location of access

How

- Data moves between main memory and secondary memory (within a node) and between main memories of different nodes
- Each data object is owned by a node
- Initial owner is the node that created object

- Ownership can change as object moves from node to node
- When a process accesses data in the shared address space, the mapping manager maps shared memory address to physical memory (local or remote).

2. ALGORITHMS FOR IMPLEMENTING DSM

Concerns

- How to keep track of the location of remote data
- How to minimize communication overhead when accessing remote data
- How to access concurrently remote data at several nodes

2.1. The Central Server Algorithm

Central server maintains all shared data

- Read request: returns data item
- Write request: updates data and returns acknowledgement message

Implementation

- A timeout is used to resend a request if acknowledgment fails
- Associated sequence numbers can be used to detect duplicate write requests
- If an application's request to access shared data fails repeatedly, a failure condition is sent to the application

Issues

- Performance and reliability

Possible solutions

- Partition shared data between several servers
- Use a mapping function to distribute/locate data

2.2. The Migration Algorithm

Operation

- Ship (migrate) entire data object (page, block) containing data item to requesting location
- Allow only one node to access a shared data at a time

Advantages

- Takes advantage of the locality of reference
- DSM can be integrated with VM at each node

Make DSM page multiple of VM page size

- A locally held shared memory can be mapped into the VM page address space
- If page not local, fault-handler migrates page and removes it from address space at remote node

To locate a remote data object:

- Use a location server
- Maintain hints at each node
- Broadcast query

Issues

- Only one node can access a data object at a time
- Thrashing can occur: to minimize it, set minimum time data object resides at a node

2.3. The Read-Replication Algorithm

Replicates data objects to multiple nodes
DSM keeps track of location of data objects
Multiple nodes can have read access or one node write access (multiple readers-one writer protocol)
After a write, all copies are invalidated or updated
DSM has to keep track of locations of all copies of data objects. Examples of implementations:

- IVY: owner node of data object knows all nodes that have copies
- PLUS: distributed linked-list tracks all nodes that have copies

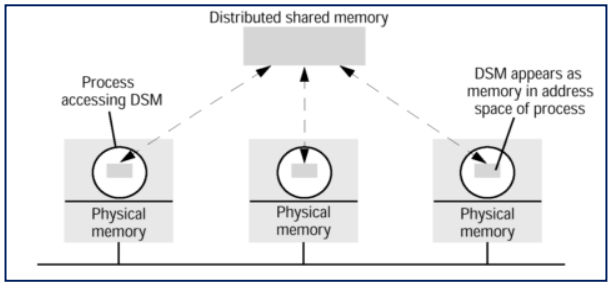
Advantage

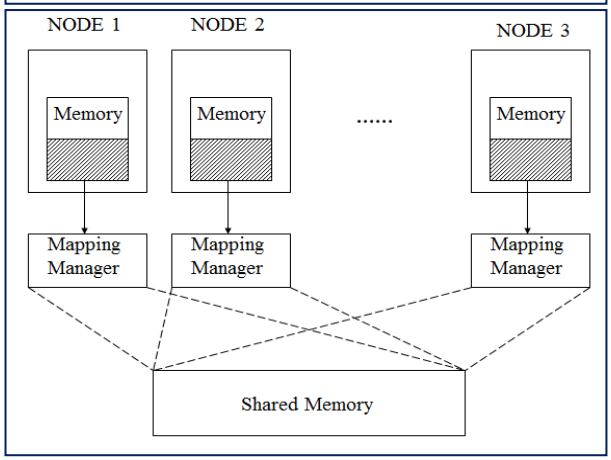
- The read-replication can lead to substantial performance improvements if the ratio of reads to writes is large

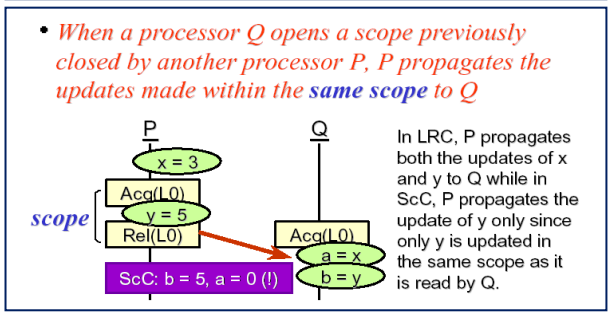
2.4. The Full–Replication Algorithm

Extension of read-replication algorithm: multiple nodes can read and multiple nodes can write (multiple-readers, multiple-writers protocol)
Issue: consistency of data for multiple writers
Solution: use of gap-free sequencer

- All writes sent to sequencer
- Sequencer assigns sequence number and sends write request to all sites that have copies
- Each node performs writes according to sequence numbers
- A gap in sequence numbers indicates a missing write request: node asks for retransmission of missing write requests.



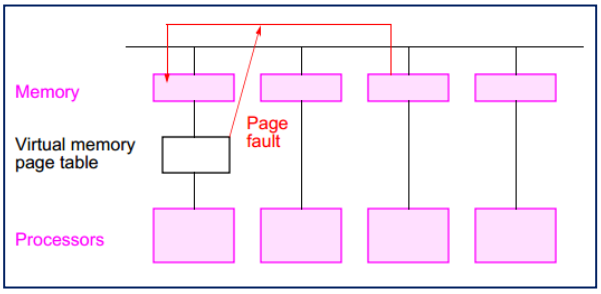




3. CONSISTENCY MODELS

A consistency model is the definition of when modifications to data may be seen at a given processor. It defines how memory will appear to a programmer by placing restrictions on the values that can be returned by a read of a memory location. A consistency model must be well understood. It determines how a programmer reasons about the correctness of programs and determines what hardware and compiler optimizations may take place.

- A consistency model is essentially a contract between the software and the memory.
- If the software agrees to obey certain rules, the memory promises to work correctly.
- If the software violates these rules, correctness of memory operation is no longer guaranteed.



3.1. Types of Consistency Models

- Strict consistency
- Sequential consistency (SC)
- Release consistency (RC)
- Scope consistency (ScC)

3.1.1. Strict Consistency

- Definition: any read to memory location x returns the value stored by the most recent write operation to x.

3.1.2. Sequential Consistency

- Definition: the result of any execution is the same as if the operations of all processors were executed in some sequential order, and the operations of each individual processor appear in this sequence in the order specified by its program.
- Any valid interleaving is acceptable behavior, but all processes must see the same sequence of memory reference.

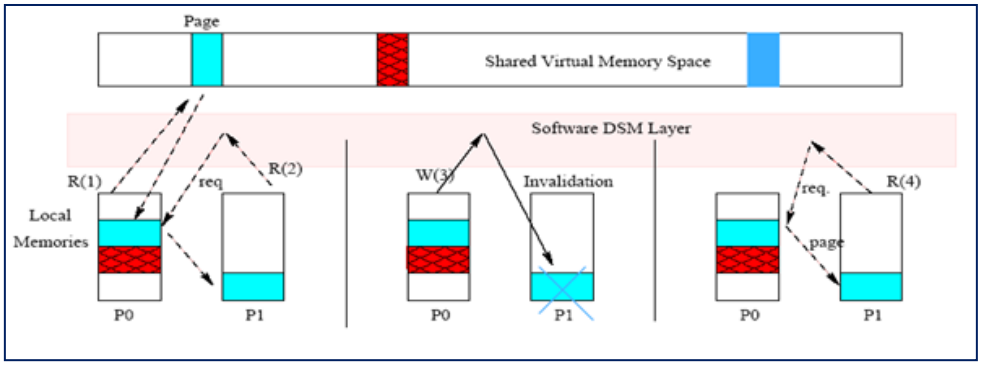
3.1.3. Release Consistency

Two types of access

- Ordinary access: read and write
- Synchronization access: acquire lock, release lock and barrier

Rules:

- Before an ordinary access to a shared variable is performed, all previous acquires done by the process must have completed successfully.
- Before a release is allowed to be performed, all previous reads and writes done by the process must have completed.



4. COHERENCE

One reaction to the cost of sequential consistency is to settle for a weaker model with well-defined properties. Coherence is an example of a weaker form of consistency. Under coherence, every process agrees on the order of write operations to the same location, but they do not necessarily agree on the ordering of write operations to different locations. We can think of coherence as sequential consistency on a location by-location basis. Coherent DSM can be implemented by taking a protocol for implementing sequential consistency and applying it separately to each unit of replicated data – for example, each page. The saving comes from the fact that accesses to two different pages are independent and need not delay one another, since the Protocol is applied separately to them. Weak consistency ◊ Dubois et al. (1988) developed the weak consistency model in an attempt to avoid the costs of sequential consistency on multiprocessors, while retaining the effect of sequential consistency. This model exploits knowledge of synchronization operations in order to relax memory consistency, while appearing to the programmer to implement sequential consistency (at least, under certain conditions that are beyond the scope of this book). For example, if the programmer uses a lock to implement a critical section, then a DSM system can assume that no other process may access the data items accessed under mutual exclusion within it. It is therefore redundant for the DSM system to propagate updates to these items until the process leaves the critical section. While items are left with 'inconsistent' values some of the time, they are not accessed at those points; the execution appears to be sequentially consistent. Adve and Hill (1990) describe a generalization of this notion called weak ordering: '(A DSM system) is weakly ordered with respect to a synchronization model if and only if it appears sequentially consistent to all software that obeys the synchronization model.' Release consistency, which is a development of weak consistency.

- Coherent memory: when value returned by read operation is the expected value (e.g., value of most recent write)
- Mechanisms that control/synchronizes accesses is needed to maintain memory coherence.

4.1. Coherence Protocol

Concerns

- How do we ensure that all replicas have the same information
- How do we ensure that nodes do not access stale data

4.1.1. Write-invalidate protocol

A write to shared data invalidates all copies except one before write executes
Invalidated copies are no longer accessible
Advantage: good performance for

- Many updates between reads
- Per node locality of reference

Disadvantage

- Invalidations sent to all nodes that have copies
- Inefficient if many nodes access same object

Examples: most DSM systems: IVY, Clouds, Dash, Memnet, Mermaid, and Mirage

4.1.2. Write-update protocol

- A write to shared data causes all copies to be updated (new value sent, instead of validation)
- More difficult to implement

5. DISTRIBUTED PAGE BASED SHARED MEMORY

The 'Page Based Distributed Shared Memory System' consists of a collection of clients or workstations connected to a server by a Local Area Network. The server contains a shared memory segment within which the distributed database is located. The shared memory segment is divided in the form of pages and hence the name 'Page Based Distributed Shared Memory System' where each page represents a table within that distributed database. In the simplest variant, each page is present on exactly one machine. A reference to a local page is done at full memory speed. An attempt to reference a page on a different machine causes a page fault, which is trapped by the software. The software then sends a message to the remote machine, which finds the needed page and sends it to the requesting process. The fault is then restarted and can now complete, which is achieved with the help of Inter Process Communication (IPC) library. In essence, this design is similar to traditional virtual memory systems: when a process touches a nonresident page, a fault occurs and the operating system fetches the page and maps it in. The difference here is that instead of getting the page from the disk, the software gets it from another processor over the network. To the user process, however, the system looks very much like a traditional multiprocessor, with multiple processes are free to read and write the shared

memory at will. All communication and synchronization is done via the memory, with no communication visible to the user process. The approach is not to share the entire address space, but only a selected portion of it, namely just those variables or data structures that needs to be used by more than one process. With respect to a distributed database system, and in this model, the shared variables represent the pages or tables within the shared memory segment. One does not think of each machine as having direct access to an ordinary memory but rather, to a collection of shared variables, giving a higher level of abstraction. This approach greatly reduces the amount of data that must be shared, but in most cases, considerable information about the shared data is available, such as their types, which helps optimize the implementation. Page-based distributed-shared memory takes a normal linear address space and allows the pages to migrate dynamically over the network on demand. Processes can access all of memory using normal read and write instructions and are not aware of when page faults or network transfers occur. Accesses to remote data are detected and protected by the memory management unit. In order to facilitate optimization, the shared variables or tables are replicated on multiple machines. Potentially, reads can be done locally without any network traffic, and writes are done using a multicopy update protocol. This protocol is widely used in distributed database system. The main purpose of this simulation is to discuss the issues in a Distributed Database System and how they can be overcome with the help of a Page Based Distributed Shared Memory System.

6. DESIGN ISSUES

Granularity: size of shared memory unit

- If DSM page size is a multiple of the local virtual memory (VM) management page size (supported by hardware), then DSM can be integrated with VM, i.e. use the VM page handling
- Advantages vs. disadvantages of using a large page size:
- (+) Exploit locality of reference
- (+) Less overhead in page transport
- (-) More contention for page by many processes
- Advantages vs. disadvantages of using a small page size
- (+) Less contention
- (+) Less false sharing (page contains two items, not shared but needed by two processes)
- (-) More page traffic
- Examples
- PLUS: page size 4 Kbytes, unit of memory access is 32-bit word
- Clouds, Munin: object is unit of shared data structure

Page replacement

- Replacement algorithm (e.g. LRU) must take into account page access modes: shared, private, read-only, writable
- Example: LRU with access modes
- Private (local) pages to be replaced before shared ones
- Private pages swapped to disk
- Shared pages sent over network to owner
- Read-only pages may be discarded (owners have a copy).

7. ADVANTAGES OF DISTRIBUTED SHARED MEMORY (DSM)

- Data sharing is implicit, hiding data movement (as opposed to 'Send'/'Receive' in message passing model)
- Passing data structures containing pointers is easier (in message passing model data moves between different address spaces)
- Moving entire object to user takes advantage of locality difference
- Less expensive to build than tightly coupled multiprocessor system: off-the-shelf hardware, no expensive interface to shared physical memory
- Very large total physical memory for all nodes: Large programs can run more efficiently
- No serial access to common bus for shared physical memory like in multiprocessor systems
- Programs written for shared memory multiprocessors can be run on DSM systems with minimum changes

8. DISADVANTAGES OF DISTRIBUTED SHARED MEMORY (DSM)

- May incur a performance penalty.
- Must provide for protection against simultaneous access to shared data (locks, etc.).
- Little programmer control over actual messages being generated.
- Performance of irregular problems in particular may be difficult.

9. METHODS OF ACHIEVING DSM

Hardware Implementation

Providing hardware support to DSM has been exploited in several ways. Some systems explore the idea of enhancing the networking capabilities of the loosely-coupledsystems. For example, the development of MemNet is based on the observation that the network is always treated as an I/O device by the communication protocols. MemNet is a shared memory local area network, based on a high-speed token ring, where the local network appears as memory in the physical address space of each processor. Capnet extends this idea to a wider domain, namely wide area networks, whereas the DASH system aims at building a scalable high performance architecture with single address space and coherent caches.

Software Implementation

During the past decade, several prototypes have been built that provide a DSM abstraction at the system level. System level implementations usually integrate the DSM as a region of virtual address space in the participating programs using the virtual memory management system of the underlying operating system. Li's shared virtual memory provides users with an interface similar to the memory address space on a multiprocessor architecture. Later, he expanded this idea to other architectures and developed a prototype called Shiva on a hypercube.

- Page based - Using the system's virtual memory
- Shared variable approach- Using routines to access
- shared variables
- Object based- Shared data within collection of objects.
- Access to shared data through object oriented discipline.

10. EXAMPLES OF DSM

Kerrighed -

Kerrighed provides several features such as a distributed shared memory with a sequential consistency model, processes migration from one cluster node to another, and to a limited extent check pointing. Kerrighed introduces a container concept: This entity is an abstraction of both files and memory.

OpenSSI

OpenSSI is an open source single-system image clustering system. It allows a collection of computers to be treated as one large system, allowing applications running on any one machine access to the resources of all the machines in the cluster.

MOSIX

MOSIX is a distributed operating system.. In a MOSIX cluster/grid there is no need to modify or to link applications with any library, to copy files or login to remote nodes, or even to assign processes to different nodes – it is all done automatically, like in an SMP.

11. CONCLUSION

Distributed Shared Memory (DSM), in Computer Architecture is a form of memory architecture where the (physically separate) memories can be addressed as one (logically shared) address space. Here, the term shared does not mean that there is a single centralized memory but shared essentially means that the address space is shared (same physical address on two processors refers to the same location in memory). Distributed Global Address Space (DGAS), is a similar term for a wide class of software and hardware implementations, in which each node of a cluster has access to shared memory in addition to each node's non-shared private memory. Software DSM systems can be implemented in an operating system (OS), or as a programming library and can be thought of as extensions of the underlying virtual memory architecture. When

implemented in the OS, such systems are transparent to the developer; which means that the underlying distributed memory is completely hidden from the users. In contrast, Software DSM systems implemented at the library or language level are not transparent and developers usually have to program differently. However, these systems offer a more portable approach to DSM system implementation. Software DSM systems also have the flexibility to organize the shared memory region in different ways. The page based approach organizes shared memory into pages of fixed size. In contrast, the object based approach organizes the shared memory region as an abstract space for storing shareable objects of variable sizes. Another commonly seen implementation uses a tuple space, in which the unit of sharing is a tuple. Shared memory architecture may involve separating memory into shared parts distributed amongst nodes and main memory; or distributing all memory between nodes. A coherence protocol, chosen in accordance with a consistency model, maintains memory coherence.

SUMMARY OF COMMUNICATION

1. This work, within the limit of available resource, has provided useful information about the distributed shared memory and its related problems.
2. It has availed researchers the opportunity to research more on the usefulness of shared memory in distributed operating systems.

FUTURE ISSUES

From understanding the concept of Distributed shared memory and its implementations, one can easily implement the same making it easy for large programs to be stored in memory and run efficiently. Future issues will focus on further improving the performance of DSM's.

DISCLOSURE STATEMENT

There is no financial support for this research work from any funding agency.

ACKNOWLEDGMENTS

We thank our guide for his timely help, giving outstanding ideas and encouragement to finish this research work successfully.

REFERENCE

1. Distritbuted shared meomory-a review by M Rasit Eskicioglu and T Anthony Marsland, University of Alberta
2. Wilson Jr. AW, Probert TH, Lane T, Fleischer B. Galactica Net: An Architecture for Distributed Shared Memory. In Proc. of the 1991 GOMAC, pages 513–516
3. Mohindra A, Ramachandran U. A Comparative Study of Distributed Shared Memory System DesignIssues. Technical Report GIT-CC-94/35, College of Computing, Georgia Institute of Technology, August 1994
4. http://en.wikipedia.org/wiki/Distributed_shared_memory
5. CSE-513: Distributed systems by Guohong Cao, Department of Computer Engineering, 310 Pond Lab.
6. Mirage: A Coherent Distributed Shared Memory Design - Fleisch, Popek– 1989
7. P.Hudak: " Memory Coherence in Shared Virtual Memory Systems - Li – 1989
8. Zwaenepoel: "Lazy Release Consistency for Software Distributed Shared Memory - Cox – 1992
9. Khandekar D. Quarks: Portable Distributed SharedMemory on Unix. Computer Systems Laboratory, University of Utah, beta ed., 1995
10. Distributed Operating System'- Andrew S. Tanenbaum
11. Pedro Souto and Eugene W. Stark, "A Distributed Shared Memory Facility for FreeBSD", Technical Conference, Anaheim, California, January 6-10, 1997